JOURNAL OF COMPUTER AND SYSTEM SCIENCES 53, 267–282 (1996)
ARTICLE NO. 0068

Manindra Agrawal*

Department of Computer Science, Indian Institute of Technology, Kanpur-208016, India

Received January 10, 1996

According to the isomorphism conjecture all NP-complete sets are polynomial-time isomorphic to each other while according to the encrypted complete set conjecture there is a p-one way function f and an NP-complete set A such that A and $f(A)$ are not polynomial-time isomorphic to each other. In this paper, these two conjectures are translated and investigated for reducibilities weaker than polynomial-time. It is shown that:

1. Relative to reductions computed by one-way logspace DTMs, both the conjectures are false.

2. Relative to reductions computed by one-way logspace NTMs, the isomorphism conjecture is true.

3. Relative to reductions computed by one-way, multi-head, oblivious logspace DTMs, the encrypted complete set conjecture is false.

4. Relative to reductions computed by constant-scan logspace DTMs, the encrypted complete set conjecture is true.

It is also shown that the complete degrees of NP under the latter two reducibilities coincide.

1. INTRODUCTION

The *isomorphism conjecture* states that all NP-complete sets are p-isomorphic to each other. It was proposed by Berman and Hartmanis [5] based on their observation that all NP-complete sets known at the time were indeed p-isomorphic to each other. Serious objections were raised against this conjecture by Joseph and Young [15]. They constructed a new type of NP-complete sets, called the *k-creative* sets, some of which did not appear to be p-isomorphic to the standard NP-complete sets. These sets (as pointed out in [20]) had the form $f(A)$ where f is a *p-one-way* function (such functions are one-one, honest, polynomial-time computable but *not* p-invertible) and A, a paddable NP-complete set. Joseph and Young argued that since p-one-way functions are not p-invertible, there may be *no* one-one, honest, p-invertible reduction of A to $f(A)$ which implies that $A \not\leqslant^p_{1,li,i} f(A)$. Based on this—and using the fact that the two NP-complete sets A and B are p-isomorphic iff $A \leqslant^p_{1,li,i} B$ and $B \leqslant^p_{1,li,i} A$ [5]—they conjectured that there is a p-one-way function f and a paddable NP-complete set A such that $f(A)$ is not p-isomorphic to A. This conjecture has been referred in the literature as the *encrypted complete set conjecture* [17].

The encrypted complete set conjecture can be equivalently stated as: there exists a p-oneway function f and a paddable NP-complete set A such that $A \not\leqslant^p_{1,li,i} f(A)$ (the equivalence follows from the above mentioned result of [5]). We shall be translating this conjecture to other reducibilities, and for some of the reducibilities that we consider, the two forms of the conjecture do not remain equivalent (for example, the reducibility 1-L). So, it becomes important as to which of these forms we choose to translate. We have opted for the second one as we feel that it reflects the intuition behind the conjecture better, viz., there is a one-way function f such that there is no one-one, honest, p-invertible reduction of A to $f(A)$, for some NP-complete set A.

It is obvious that not both of these conjectures, viz., the isomorphism and the encrypted complete set, can be true simultaneously. So the question is—which of the two conjectures, if any, is true? As both the conjectures imply, amongst other things, $P \neq NP$, it is a difficult question to answer. This has led to the relocation of the conjectures to classes that are provably larger than P. e.g., EXP, NEXP. As a proof of the encrypted complete set conjecture for these classes would still imply $P \neq NP$, it remains a difficult problem. However, attempts to *disprove* it, and hence prove the isomorphism conjecture, have not been successful either for any of these classes. There have only been some partial collapsing results—the $\leqslant^p_m$-complete degree for EXP collapses to $\leqslant^p_{1,li}$-complete degree [4, 23, 9]; the $\leqslant^p_m$-complete degree for NEXP collapses to $\leqslant^p_l$-complete degree [9]. These failures are not surprising in view of certain relativization results—it has been shown [18] that relative to a *random* oracle (see [19] for definition), the encrypted complete set conjecture holds for NP, EXP, NEXP etc. On the other hand, it has been recently shown [7] that relative to a *sp-generic* oracle (see [7] for definition) the isomorphism conjecture is true for all these classes. A good survey of the results concerning these two conjectures can be found in [17].

* E-mail address: manindra@iitk.ernet.in.

0022-0000/96 $18.00

As the conjectures, even when relocated to the higher classes, have proved to be difficult to settle, one may try another approach by relocating them to *weaker reducibilities*. This leads us to the following generalization of the conjectures—for any class of reductions $\mathscr{F}(r)$, let the *r-isomorphism conjecture* be that all $\leqslant_m^r$-complete sets for NP are r-isomorphic; and the *r-encrypted complete set conjecture* be that there exists a *r-one-way function* f (i.e., f is a one-one, honest, function in $\mathscr{F}(r)$ such that its inverse does not belong to $\mathscr{F}(r)$) and a $\leqslant_m^r$-complete set A for NP such that $A \not\leqslant_{1,li,i}^p f(A)$. These conjectures have been investigated for various reducibilities weaker than polynomial time [11, 3, 1], however, again, no answers have been found. In this paper, we provide, for the first time, answers to the two conjectures for several weak reducibilities. All the results below hold for any class closed under log-lin reductions, we restrict ourselves to NP as it is the most interesting one.

We first consider 1-*L reductions*, the class of functions computed by logspace bounded DTMs with a one-way input head. Complete degrees for these reductions are non-trivial—it has been shown [12] that all *natural* NP-complete sets are complete under 1-L reductions as well (though one can easily construct an NP-complete set which is not $\leqslant_m^{1-L}$-complete [12]). The structure of complete degrees under 1-L reductions has been investigated before [3, 9, 13, 6, 1], we improve on all the earlier results. We show that even though 1-L-one-way functions exist, the $\leqslant_m^{1-L}$-complete degree of NP collapses to the $\leqslant_{1,li,i}^{1-L}$-complete degree. Thus the 1-L-encrypted complete set conjecture is false. With such a strong collapse of $\leqslant_m^{1-L}$-complete degree, one may expect the 1-L-isomorphism conjecture to be true, however, it turns out that this conjecture too is false. Nevertheless, by generalizing the class of reductions to 1-*NL*, functions computed by logspace bounded NTMs with a one-way input head, we show that the 1-NL-isomorphism conjecture is true.

Next, we consider another generalization of 1-L reductions: 1-*omL reductions*, functions computed by logspace bounded *oblivious* DTMs with *multiple* one-way input-heads. Such TMs can compute several p-one-way functions (defined by using the computation of TMs recognizing languages in UP-P), and therefore, are fairly powerful. We show that even the $\leqslant_m^{1-omL}$-complete degree of NP collapses to the $\leqslant_{1,li,i}^{1-omL}$-complete degree and therefore, the 1-omL-encrypted complete set conjecture too is false. The last class of reductions we consider are *c-L reductions*, computed by logspace bounded DTMs with their input head allowed a fixed constant number of left-to-right scans of the input tape. These reductions too are a generalization of 1-L reductions and form a proper subclass of 1-omL reductions. We show that the *c-L*-encrypted complete set conjecture is true. The last two reducibilities exhibit another interesting property: $\leqslant_m^{1-omL}$-and $\leqslant_m^{c-L}$-complete degrees for NP *coincide*, and all such complete sets are complete under one-one, length-increasing, *c-L* reductions with their inverses being 1-omL functions. This provides probably the first example of a reducibility (1-omL), the complete sets under which are also complete under a *strictly weaker* reducibility (*c-L*).

We begin the paper by first introducing the basic terminology the we use (section 2) and then in section 3, we define the notion of forgetful TMs, which plays a crucial part in almost all our proofs. We then move on to giving the results for the four reducibilities in sections 4, 5, 6 and 7. In section 8 we discuss the implications of these results and compare our technique with the earlier ones. Finally, we list some open questions in section 9.

2. PRELIMINARIES

The strings are over $\Sigma = \{0, 1\}$. To avoid confusion between strings and numbers, we write 0's and 1's of a string in boldface. For a string s, $|s|$ denotes its length. For a finite set of strings S, $\|S\|$ denotes the number of strings in S. Set Σ^n denotes the set of all strings of length n. For any string s and for any number i, $1 \leqslant i \leqslant |s|$, $s[i]$ denotes the ith bit of s.

Our model of computation is Turing machines with a read-only input tape, a write-only output tape and a read-write work tape. We shall assume, without loss of generality, that a TM has a unique initial state and it halts only after placing the input head(s) on the cell immediately to the right of the input string. Further, any time it moves the input head (some input head if there are more than one) the TM makes a special mark on the work tape, then erases it, and only then reads the new input bit. This property will be useful in identifying certain stages of the TM.

Function f is a *log-lin* function [21] if it can be computed by logspace bounded DTMs and for all x: $|f(x)| = O(|x|)$.

For a resource bound r on TMs, we denote by $\mathscr{F}(r)$ the class of total functions computed by TMs within the resource bound of r. For the class of functions $\mathscr{F}(r)$, we say that f is an *r-computable function*, or simply, an *r function*, if $f \in \mathscr{F}(r)$; and f is *r-invertible* if there is a function $g \in \mathscr{F}(r)$ such that $g(f(x)) = x$ for every x. We say that the set $A \leqslant_m^r (\leqslant_{1,li}^r; \leqslant_{1,li,i}^r; \leqslant_{1,qli,i}^r) B$ if there is a many-one (one-one, length-increasing; one-one, length-increasing and r-invertible; one-one, quadratic length-increasing and r-invertible) r-computable function f reducing A to B. The set A is $\leqslant_m^r$-*hard* for class $\mathscr{C}$ if for every $B \in \mathscr{C}$, $B \leqslant_m^r A$. The set A is $\leqslant_m^r$-*complete* for class $\mathscr{C}$ if A is $\leqslant_m^r$-hard for $\mathscr{C}$ and $A \in \mathscr{C}$. For the class NP, a NP-*complete* set is a $\leqslant_m^p$-complete set for NP. The $\leqslant_m^r$-*complete degree* of $\mathscr{C}$ is defined to be the class of all $\leqslant_m^r$-complete sets for $\mathscr{C}$. Similarly, one defines these notions for $\leqslant_{1,li}^r$, $\leqslant_{1,li,i}^r$ and $\leqslant_{1,qli,i}^r$ reductions. We say that the set A is *r-isomorphic* to the set B if there exists a bijection f between A and B with both f and f^{-1} being r-computable.

A 1-*L TM* is a deterministic Turing machine with a read-only input tape, a write-only output tape and a logspace-bounded work tape such that its input head is *one-way*, i.e., it moves from left to right only. Further, at the beginning of the computation, $\mathbf{1}^{\lceil \log n \rceil}$ is written on the work tape, where n is the length of the input. These functions were first defined in [12] for studying complete sets for DLOG. The class of $\leqslant_m^{1-L}$-complete sets for NP is a fairly large one: it was shown in [12] that all known *natural* NP-complete sets are $\leqslant_m^{1-L}$-complete as well.

It is worth noting at this point that 1-L functions are not closed under composition as 1-L TMs need $\mathbf{1}^{\lceil \log n \rceil}$ written on the work tape at the beginning of the computation [3]. Nevertheless, the notion of our interest, viz., $\leqslant_m^{1-L}$-complete degrees, is well defined.

A 1-*NL TM* is a non-deterministic Turing machine with the rest of the conditions being the same as for a 1-L TM (note that for these TMs also the workspace is marked off in advance). Class $\mathscr{F}$(1-NL) contains total functions that are computed by 1-NL TMs that output the *same* string on all accepting paths. These functions are closed under composition as a 1-NL TM can guess the length of the input and verify it later on.

A k-*L TM*, $k > 0$, is a deterministic Turing machine with the same conditions as for a 1-L TM except that its input head is allowed a maximum of k left-to-right scans of the input tape. A *c-L TM* is one that is a k-L TM for some $k > 0$. The class of total functions computed by c-L TMs is $\mathscr{F}(\text{c-L}) = \bigcup_{k>0} \mathscr{F}(k\text{-L})$. It is clear that the class $\mathscr{F}$(c-L) is closed under composition (composition of a k_1-L and k_2-L function is a $k_2 \cdot (k_1 + 1)$-L function).

A 1-*mL TM* is a deterministic Turing machine with a read-only input tape, a write-only output tape and a logspace-bounded work tape. It may have *more* than one input heads with all of them being one-way. The class $\mathscr{F}$(1-mL) is closed under composition. We shall mainly be interested in a subclass of these functions computed by 1-mL TMs that are oblivious. Recall that an *oblivious* TM is one whose input head(s) movement depends only on the length of the input. We refer to the oblivious 1-mL TMs as 1-*omL TMs*. The class $\mathscr{F}$(1-omL) is not closed under composition. An example is—$f(x) = y$ where $x = \mathbf{1}^j\mathbf{0}y\mathbf{01}^k$ for $j, k \geqslant 0$; $g(y) = \mathbf{1}$ if $y = zz$, $g(y) = \mathbf{0}$ otherwise. Both f and g are 1-omL functions but their composition $g \circ f$ is not. We leave the proof to the reader.

3. FORGETFUL 1-L TMs

In this section, we define the notion of forgetful TMs, which plays an important role in all our collapse results. Throughout the section, we shall be dealing with Turing machines computing 1-L functions. We begin by defining the notion of a configuration of such TMs.

DEFINITION 3.1. Let M be a 1-L TM. A *configuration of M of size n* is a partial ID of M on an input of size n. It is written as a 5-tuple $\langle st, in, out, wk, tape \rangle$ where st denotes the state of M; in, out and wk denote respectively the input head, output head and work tape head positions; and tape denotes the contents of the work tape. We refer to the initial configuration of M of size n as C^n_{init} (C^n_{init} may be taken to be $\langle q_0, 1, 1, 1, \mathbf{1}^{\lceil \log n \rceil} \rangle$ where q_0 is the start state).

Let $config(M, n)$ denote the set of all configurations of a 1-L TM M of size n. The number of configurations in $config(M, n)$ is bounded by a polynomial in n as M is a logspace TM. Let this bounding polynomial be denoted by q_M.

Recall that a TM in our model, after it moves the input head any time, makes a special mark on the work tape, then erases it, and only then reads the new input bit.

DEFINITION 3.2. A *transit* configuration of size n of a 1-L TM is either (1) the initial configuration of size n of the TM, (2) a final configuration of size n of the TM, or (3) a configuration of the TM with the special mark on the work tape.

Let $i(C)$ denote the position of the input head in the configuration C. The following definition will be frequently used below.

DEFINITION 3.3. A 1-L TM M *moves from configuration C to D on reading s* if $i(D) = i(C) + |s|$, D is a transit configuration, and the TM M when started on the configuration C with the string s written on the bit positions $i(C)$ through $i(D) - 1$ of the input, ends in configuration D.

In the above definition the usefulness of transit configurations becomes apparent—the string s is written in the bit positions up to $i(D) - 1$ only and so the TM, while moving from C to D, should not read the bit at the position $i(D)$ for the definition to be robust.

We now define the notion of forgetful TMs.

DEFINITION 3.4. A 1-L TM is *forgetful* if for every n, the sequence of transit configurations the TM passes through on any input of size n, is identical. A function computed by a forgetful 1-L TM is called a *forgetful* 1-L function.

A forgetful 1-L TM, when it is in a transit configuration, does not "remember" the value of any bit of the input, except perhaps its length. This severely restricts the power of the TM. However, we now show that for any class that is closed under log-lin reductions, any set that is complete for the class under 1-L reductions is also complete under reductions computed by forgetful 1-L TMs.

THEOREM 3.5. *For any class C closed under log-lin reductions, any $\leqslant_m^{1-L}$-hard set for C is also hard under forgetful 1-L reductions.*

Proof. Let A be a $\leqslant_m^{1-L}$-hard set for $\mathscr{C}$ and B be an arbitrary set in $\mathscr{C}$, $B \neq \varnothing, \Sigma^*$. We shall exhibit a reduction of B to A computed by a forgetful 1-L TM. We first define a set D as accepted by the following procedure.

Input y. Let $y = w\mathbf{01}^b\mathbf{0}^r$ for some $r \geqslant 0$. If $2b$ does not divide $|w|$ then reject. Otherwise, let $w = w_1 w_2 \cdots w_n$ where $|w_i| = 2b$ for $1 \leqslant i \leqslant n$. Define a string x, $|x| = n$ such that $x[i] = \mathbf{1}$ if $w_i = uu$ for some string u, $\mathbf{0}$ otherwise. Accept iff $x \in B$.

Set D can clearly be reduced to B via a log-lin reduction and therefore, $D \in \mathscr{C}$. Let f be a 1-L reduction of D to A computed by the TM M. We define a reduction, g, of B to D, based on the TM M, as given by the following stage-wise procedure.

Input x, with $|x| = n$.

Stage 0: Let $m = d_0 \cdot 2^{2\lceil \log n \rceil}$ and $b = \lceil \log q_M(m) \rceil + 1$ (recall that $q_M(n)$ is the bound on the number of possible configurations of M of size n) where d_0 is the smallest number such that $m \geqslant (2n+1)\, b + 1$. Let C_0 be the initial configuration of M of size m, i e, C_{init}^m.

Stage j, $1 \leqslant j \leqslant n$: Find the smallest (in the lexicographic order) transit configuration C of M (of size m) with $i(C) = 2bj + 1$; and the smallest (again, in the lexicographic order) two strings u and v, $u \neq v$ and $|u| = |v| = b$, such that M moves from C_{j-1} to C on reading either of the strings uu and vu. Let $C_j = C$ and $w_j = uu$ if $x[j] = \mathbf{1}$, vu otherwise.

Stage $n+1$: Let $r = m - (2n+1)\, b - 1$. Output the string $w_1 w_2 \cdots w_n \mathbf{01}^b\mathbf{0}^r$.

We now show that the configuration C and strings u and v, as required in the Stage j of the above procedure, can always be found. Since $|u| = |v| = b$, there exist 2^b such strings. As the TM M can have at most $q_M(m)$ transit configurations of size m, it follows from the Pigeon Hole principle that there must be at least $2^b/q_M(m)$ strings such that the TM M ends up in the same transit configuration (when started from the configuration C_{j-1}) on reading any of these strings. By the choice of b, $2^b/q_M(m) \geqslant 2$. So there are at least two such strings.

The entire procedure can be carried out by a logspace TM as the length of the configurations and b is bounded by $O(\log n)$. The input head of the TM needs only be one-way as it needs to scan the input x only once from left to right. Therefore, the function g is a 1-L function. It is clear, from the definition of the set D that g is a reduction of B to D. Therefore, $h = f \circ g$ is a reduction of B to A. Function h can be computed by the following forgetful 1-L TM.

On input x, $|x| = n$, the TM first computes $\mathbf{1}^{\lceil \log |g(x)| \rceil} = \mathbf{1}^{\lceil \log d_0 \rceil + 2\lceil \log n \rceil}$ using the string $\mathbf{1}^{\lceil \log n \rceil}$ written on the worktape. Now it simulates the TM M on $g(x)$ while computing $g(x)$ in parallel. The TM has, initially, the initial configuration of M on input $g(x)$, say C_0, written on the worktape. For every j, $1 \leqslant j \leqslant n$, the TM, before reading the jth bit of the input, takes the configuration C_{j-1} of M and, using the Stage j of the procedure above, computes the two-strings u and v and the configuration C_j. Now it scans the j^{th} input bit and computes the string w_j with $w_j = uu$ if $x[j] = \mathbf{1}$, vu otherwise. Then, it continues with the simulation of M on w_j. At the end of this simulation, M will end up in the configuration C_j no matter which of the two values w_j takes. After scanning all the bits of the input, the TM continues the simulation of M on the string $\mathbf{01}^b\mathbf{0}^r$ where $r = m - (2n+1)b - 1$ as above (the TM can compute this as now it knows n).

It is easy to see that the above TM is a forgetful 1-L TM computing the function h. ∎

4. 1-L REDUCTIONS

In this section, we consider complete degrees under 1-L reductions. The isomorphism question for such degrees (and for ones complete under 1-NL reductions) has been considered earlier too. Allender [3] showed that $\leqslant_m^{1-L}$-complete sets for PSPACE and EXP are p-isomorphic; Ganesan and Homer [9] showed the same result for the class NEXP; Hemachandra and Hoene [13] showed that $\leqslant_m^{1-L}$-(or, $\leqslant_m^{1-NL}$-) complete sets for non-deterministic space classes above NLOG are also $\leqslant_{1,\,li}^{1-L}$-(resp., $\leqslant_{1,\,li}^{1-NL}$-) complete and hence NLOG-isomorphic; Hoene and Burtschick [6] showed that $\leqslant_m^{1-L}$-complete sets for PSPACE are not 1-L-isomorphic; and finally Agrawal and Biswas [1] showed that $\leqslant_m^{1-L}$-complete sets for classes closed under log-lin reductions are p-isomorphic. Our results, in this and the next section, generalize all the previous ones, both for 1-L and 1-NL reductions.

The section is divided into three subsections. In the first one, we prove our main result, viz., that for any class closed under log-lin reductions, the $\leqslant_m^{1-L}$-complete degree collapses to the $\leqslant_{1,\,qli,\,i}^{1-L}$-complete degree. In the next subsection we show that there exist 1-L-one-way functions and in the last one we show that the 1-L-isomorphism conjecture does not hold.

4.1. *Collapse of Complete Degrees*

In this subsection, we show that all $\leqslant_m^{1-L}$-hard sets for any class $\mathscr{C}$ closed under log-lin reductions, are also $\leqslant_{1,\,qli,\,i}^{1-L}$-hard.

As remarked in the section 2, 1-L functions are not closed under composition. However, we shall need to compose 1-L functions several times and shall require that the composition remains a 1-L function. To achieve this, we define a restricted type of 1-L functions whose compositions with 1-L functions remain 1-L functions. Say that a 1-L function g is *length-restricted* if for every x, $\lceil \log |g(x)| \rceil = \lceil \log |g(\mathbf{1}^{2^{\lceil \log |x| \rceil}})| \rceil$. Now, we have the following proposition.

LEMMA 4.1. *Let f be a 1-L function and g a length-restricted 1-L function. Then, $f \circ g$ is also a 1-L function. Further, if both f and g are forgetful, then $f \circ g$ is also forgetful.*

Proof. Let M_1 and M_2 be 1-L TMs computing f and g respectively. A 1-L TM M can compute $f \circ g$ by simulating M_2 on input x and M_1 on its output in parallel. However, to start the computation of M_1, it needs to have $\mathbf{1}^{\lceil \log |g(x)| \rceil}$ written on the worktape. Since g is length-restricted, this string can be computed without scanning the input. The TM M just runs M_2 on the input $\mathbf{1}^{\lceil \log |x| \rceil}$ and calculates the length of the output $(= |g(\mathbf{1}^{2^{\lceil \log |x| \rceil}})|)$. Using this, $\lceil \log |g(x)| \rceil$ can be easily computed.

If both M_1 and M_2 are forgetful, then M would also be forgetful as M needs to store only the configurations of these two TMs on the worktape (besides some input independent information). ∎

THEOREM 4.2. *For any class $\mathscr{C}$ closed under log-lin reductions, every $\leqslant_m^{1-L}$-hard set for $\mathscr{C}$ is also $\leqslant_{1,\,qli,\,i}^{1-L}$-hard.*

Proof. Let A be a $\leqslant_m^{1-L}$-hard set for $\mathscr{C}$. We prove the theorem in two stages. In the first stage, we show that A is hard under forgetful 1-L reductions that are size-nondecreasing and one-one on Σ^n for every $n \geqslant 1$. And in the second stage, we improve this to length-squaring, one-one, and invertible reductions.

Stage 1: Let $B \in \mathscr{C}$, $B \neq \Sigma^*, \varnothing$. Define a set D as accepted by the following procedure.

> Input y. If $|y|$ is not even, then reject. Otherwise, let $y = xs$ with $|x| = |s| = |y|/2$. If $s \in \mathbf{1}^*$ then accept iff $x \in B$. If $s = \mathbf{1}^{i-1}\mathbf{0}\mathbf{1}^{|s|-i}$ then accept iff $x[i] = \mathbf{1}$. Otherwise, reject.

It is easy to see that $D \leqslant_m^{log\text{-}lin} B$, and therefore, $D \in \mathscr{C}$. So, by Theorem 3.5, there exists a reduction, say f, of D to A computed by a forgetful 1-L TM, say M. Function $g(x) = x\mathbf{1}^{|x|}$ is a reduction of B to D. So function $h = f \circ g$ is a reduction of B to A. By Lemma 4.1 it follows that it is a forgetful 1-L function as g is length-restricted and forgetful.

CLAIM 4.2.1. *Function h is size-nondecreasing and one-one on Σ^n for every $n \geqslant 1$.*

Proof of Claim 4.2.1. Suppose that h is not one-one on Σ^n for some $n \geqslant 1$. Let $x, y \in \Sigma^n$, $x \neq y$, be such that $h(x) = h(y)$ and let $x[i] \neq y[i]$ for some i. Note that $g(x) = x\mathbf{1}^n$ and $g(y) = y\mathbf{1}^n$. Consider the strings $x' = x\mathbf{1}^{i-1}\mathbf{0}\mathbf{1}^{n-i}$ and $y' = y\mathbf{1}^{i-1}\mathbf{0}\mathbf{1}^{n-i}$. We argue that $f(x') = f(y')$. Let $f(x') = u_1u_2$ and $f(y') = v_1v_2$ where u_1 (resp. v_1) is the output of M while scanning the first half of the string x' (resp. y'). Since the first half of x' is identical to that of $g(x)$, u_1 must equal the output of M while scanning the first half of $g(x)$. Similarly, v_1 must equal the output of M while scanning the first half of $g(y)$. Since $g(x) = g(y)$, and the TM M outputs an equal number of bits while scanning the first halves of $g(x)$ or $g(y)$ (follows from the fact that M is forgetful), it follows that $u_1 = v_1$. Now, the second halves of x' and y' are identical. Therefore, since M is forgetful, the output of M while scanning this half must be the same for both the strings. Thus, $u_2 = v_2$ and so, $f(x') = f(y')$. However, this contradicts the fact that f is a reduction of D to A as exactly one of the two strings x' and y' belong to D.

Therefore, h must be one-one on Σ^n for every n. Since a forgetful TM halts in the same configuration for all strings of size n, it follows that for every two strings x and y in Σ^n, $|h(x)| = |h(y)|$. This implies that $|h(x)| \geqslant |x|$ for every x. ∎

Stage 2: From the Stage 1, we have that A is complete under forgetful 1-L reductions that are one-one on Σ^n for every $n \geqslant 1$. We have to do a little more work to make them one-one everywhere. Again, let $B \in \mathscr{C}$, $B \neq \Sigma^*, \varnothing$. We define a set D',

$$D' = \{\mathbf{0}^k\mathbf{1}x\mathbf{1}\mathbf{0}^j \mid k, j \geqslant 0 \wedge x \in B\}.$$

It is easy to see that $D' \leqslant_m^{log\text{-}lin} B$ and therefore, $D' \in \mathscr{C}$. Let f' be a forgetful 1-L reduction, as constructed in the Stage 1, of D' to A. Let $|f'(x)| \leqslant c \cdot |x|^c$, for some constant $c > 0$. Let $r(l) = (2c)^l$ for $l \geqslant 0$. Define a function g', reducing B to D' as: $g'(x) = \mathbf{0}^k\mathbf{1}x\mathbf{1}\mathbf{0}^j$ where $k = \lceil \log |x| \rceil$, $j = 2^{2r(l)} - \lceil \log |x| \rceil - |x| - 2$, and $l = min_m(r(m) \geqslant \lceil \log |x| \rceil)$. Therefore, $|g'(x)| = 2^{2r(l)}$. Function g' is a 1-L function—on input x, the 1-L TM computing g' calculates $k = \lceil \log |x| \rceil$ and outputs $\mathbf{0}^k\mathbf{1}x$. Now it calculates $r(1), r(2), \ldots$ till an $r(l)$ is obtained with $r(l) \geqslant \lceil \log |x| \rceil$; then it computes $j = 2^{2r(l)} - \lceil \log |x| \rceil - |x| - 2$ and outputs $\mathbf{1}\mathbf{0}^j$. This function maps strings of length between $2r^{(l-1)} + 1$ and $2^{r(l)}$ to strings of length $2^{2r(l)}$ and therefore, is quadratic length-increasing. Note that in the computation of $g'(x)$, the 1-L TM knows only the number $\lceil \log |x| \rceil$ before scanning x and therefore to make the output length a power of two, it needs to pad $\mathbf{0}$'s at the end as well. Of course, this could have been done by padding $\mathbf{0}$'s only at the end, however, to strengthen the isomorphism between $\leqslant_m^{1-L}$ complete sets, it is necessary to pad $\lceil \log |x| \rceil$ many $\mathbf{0}$'s at the beginning (see Theorem 4.6 below).

Let $h' = f' \circ g'$. We show that h' is the required reduction of B to A. It is a forgetful 1-L function as g' is a forgetful and length-restricted 1-L function.

CLAIM 4.2.2. Function h' is length-squaring and one-one.

Proof of Claim 4.2.2. Since g' is length-squaring and f' is length-nondecreasing, h' is clearly length-squaring. For any two strings x and y, $x \neq y$, if $|g'(x)| = |g'(y)|$, then $h'(x) \neq h'(y)$ as g' is one-one and f' is one-one on strings of equal length. On the other hand, if $|g'(x)| > |g'(y)|$, then, letting $|g'(y)| = 2^{2r(l)}$, we have that $|h'(y)| = |f'(2^{2\cdot(2c)^l})| \leqslant c \cdot 2^{(2c)^{l+1}}$ (by the bound on the length of f') $\leqslant 2^{2(2c)^{l+1}-c} = 2^{2r(l+1)-c}$ (by the definition of r) $\leqslant |g'(x)|/2^c$ (since $|g'(x)| \geqslant 2^{2r(l+1)}$) $< |h'(x)|$ (since f' is size-nondecreasing and $c > 0$). Therefore, h' is one-one. ∎

To complete the proof, we also need to show that h' is 1-L-invertible. We achieve this in two steps. First, we give a 1-L TM that computes the inverse of f' on the range of h', and then a 1-L TM that computes the inverse of g'. A composition of these two TMs would give us the required TM.

Let f' be computed by the forgetful TM M'. A 1-L TM, say M_1, executing the following procedure, computes the inverse of f' whenever the input is in the range of h', rejects otherwise.

> Input z. Compute an l such that, letting $n = 2^{2r(l)}$, if $w = f'(\mathbf{1}^n)$ then $\mathbf{1}^{\lceil \log |w| \rceil} = \mathbf{1}^{\lceil \log |z| \rceil}$. There will be *at most* one such l since $|h'(x)| \leqslant 1/2 \cdot |h'(y)|$ if $|g'(x)| < |g'(y)|$ (as argued above). If there is no such l then halt in a rejecting state.
>
> Otherwise, start simulating the TM M' on the strings of size n (if $h'^{-1}(z)$ is defined then $|f^{-1}(z)|$ must be n). Compute the output of M'while it scans the first bit of any such input. There would be two outputs corresponding to the cases when the first bit is **1** and when it is **0**. These two outputs must be of the same length and different since M' is forgetful and f' is one-one on equal length strings. Therefore, at most one of these outputs can be a prefix of z. If neither is, then halt in a rejecting state. Otherwise, output the bit whose output is a prefix. Repeat the same procedure for all the subsequent bits input to M'. At the end, if the procedure has been successfully carried out for all the bits, then halt in an accepting state.

The following procedure computes the inverse of g'.

> Input y. Let $y = \mathbf{0}^k\mathbf{1}x\mathbf{10}^j$ for some $k, j \geqslant 0$. Output x. Now, check if $k = \lceil \log |x| \rceil$ and $|y| = 2^{2r(l)}$ for the smallest l such that $r(l) \geqslant \lceil \log |x| \rceil$. If yes, then halt in an accepting state otherwise halt in a rejecting state.

A 1-L TM, say M_2, that carries out the above procedure, after scanning the leading zeroes and the first one, outputs x using the following strategy: the moment a **1** is encountered count the number of contiguous zeroes immediately after it, say i, and output $\mathbf{10}^i$ only if there is more input.

The TM that computes the inverse of h', on input z, simulates the above two TMs, M_1 on z and M_2 on the output of M_1. It outputs the output of M_2, accepts if both the TMs accept, rejects otherwise. Since both M_1 and M_2 are 1-L TMs and M_2 does not require the length of the input at the beginning of its computation, M' is also a 1-L TM.

Therefore, $B \leqslant \leqslant^{1-L}_{1,\,qli,\,i} A$ via h'. ∎

COROLLARY 4.3. *For any class $\mathscr{C}$ closed under log-lin reductions, $\leqslant^{1-L}_m$-complete degree of $\mathscr{C}$ collapses to a $\leqslant^{1-L}_{1,\,qli,\,i}$-complete degree.*

COROLLARY 4.4. *For any class $\mathscr{C}$ closed under log-lin reductions, if A is a $\leqslant^{1-L}_m$-hard set for $\mathscr{C}$ and t is a one-one, 1-L function then $t(A)$ is also $\leqslant^{1-L}_m$-hard.*

Proof. As 1-L functions are not closed under composition in general, it is conceivable that $t(A)$ is not $\leqslant^{1-L}_m$-hard for $\mathscr{C}$ for some $\leqslant^{1-L}_m$-hard set A and a one-one, 1-L function t. However, it is easy to check that the reduction h' constructed in the proof of the above theorem is length-restricted (besides being forgetful). Therefore, by Lemma 4.1, $t \circ h'$ is also a 1-L function implying that $t(A)$ is also $\leqslant^{1-L}_m$-hard. ∎

The above corollary along with the Theorem 4.2 implies that,

COROLLARY 4.5. *The 1-L-encrypted complete set conjecture is false.*

It trivially follows, from the above theorem and a result in [11], that the $\leqslant^{1-L}_m$-complete sets for $\mathscr{C}$ are logspace-isomorphic. However, one can do better than this and show that these sets are 2-*L-isomorphic*, where 2-L functions are computed by logspace TMs that are allowed two left-to-right scans of the input.

THEOREM 4.6. *For any class $\mathscr{C}$ closed under log-lin reductions, the $\leqslant^{1-L}_m$-complete sets for $\mathscr{C}$ are 2-L-isomorphic.*

Proof. In [11], it was shown that all sets complete under $\leqslant^{log}_{1,\,qli,\,i}$-reductions are logspace-isomorphic. We shall proceed along exactly the same lines. The construction in [11] is inspired from the one for polynomial-time reducibilities given in [5] which, in turn, is essentially the Cantor–Schröder–Bernstein construction of the isomorphism between two sets. We first give the construction of [11] and then mention the modifications needed to make it work for 1-L reductions.

Let A and B be two sets with $A \leqslant^{log}_{1,\,qli,\,i} B$ via u and $B \leqslant^{log}_{1,\,qli,\,i} A$ via v. For any x, define the *inverse chain* at x to be the sequence $x_0, x_1, x_2, ..., x_l$ where $x_0 = x$, $x_i = v^{-1}(x_{i-1})$ if i is odd, $u^{-1}(x_{i-1})$ if i is even, for $1 \leqslant i \leqslant l$ and

$v^{-1}(x_l)(u^{-1}(x_l))$ is not defined if l is even (odd). Number l is called the length of the chain.

Define function t as: $t(x) = u(x)$ if the length of the inverse chain at x is even; $v^{-1}(x)$ otherwise. It is easy to see that t is an isomorphism between A and B. Since both u and v are length squaring and their inverses are computable in logspace, it follows that the length of the inverse chain at x can be computed in logspace by an interleaved simulation of all inverses, and therefore, t can be computed in logspace.

In our case, u and v are one-one, length-squaring, 1-L-invertible, 1-L functions. Define t as before. We show that the length of the inverse chain at x can be computed in a single pass over the input and therefore t can be computed in two passes (use the second pass to compute $u(x)$ or $v^{-1}(x)$ depending on whether the length is even or odd). In a similar manner t^{-1} can also be computed.

We have, by the proof of the Theorem 4.2, two 1-L TMs M_u and M_v that, on input x, output $u^{-1}(x)$ and $v^{-1}(x)$ respectively if they are defined and halt in an accepting state. Otherwise the TMs halt in a rejecting state. The following procedure to compute the length of the chain suggests itself-on input x, start the computation of M_v on x; the moment some output appears, start the computation of M_u on it; and so on for all the intermediate strings. If at any point, some computation ends in a rejecting state (the inverse is not defined), abort all the computations started after it. Eventually, the first k computations will end in an accepting state and the $(k+1)$th in a rejecting state for some $k \geqslant 0$ and then k will be the length of the chain. However, this procedure does not work as to be able to compute $v^{-1}(x_i)$ or $u^{-1}(x_i)$ in a single pass at any intermediate string x_i, one needs $\mathbf{1}^{\lceil \log |x_i| \rceil}$ also written on the tape. It is for this reason that the function g' was defined in a somewhat complicated way in the proof of the Theorem 4.2. Assume that $u = f'_1 \circ g'_1$ and $v = f'_2 \circ g'_2$ as constructed in the Stage 2 of the proof. The above procedure is modified in the following way.

On input x, start the computation of v^{-1} on x and count the number of leading zeros in the (intermediate) string $(f'_2)^{-1}(x)$. If the inverse is defined on x, there must be exactly $\lceil \log |x_1| \rceil$ such zeroes. Continue with the computation, it must now output x_1. Simultaneously, start the computation of $u^{-1}(x_1)$; it can be computed properly as we know $\lceil \log |x_1| \rceil$. In this way, we can know the length of all intermediate strings before their computation begins. The rest of the procedure remains the same. Thus the entire computation can be performed in a single pass over the input. ∎

In [12], it was shown that all natural NP-complete sets are $\leqslant_m^{1-L}$ complete as well. It follows that all these sets are 2-L-isomorphic. This is an improvement on the result that they are all logspace-isomorphic [11] while it does not compare with a recent result that they are first-order-isomorphic [2].

4.2. *Existence of One-Way Functions*

It is fairly straightforward to see that there are 1-L-one-way functions. Define $f(bx) = xb$ where $b \in \Sigma$. Clearly f is a 1-L function. Moreover, it is a one-one and onto function. Now we show that f^{-1} is not a 1-L function.

PROPOSITION 4.7. *Function h, $h(xb) = bx$, $b \in \Sigma$, is not a 1-L function.*

Proof Sketch. Any DTM computing h must read the last bit of the input before outputting any bit. Any such DTM working within logspace will "forget" most of the string x, for large enough x, by the time it reaches the end of the input and therefore to output $h(xb)$ correctly, it must scan the input once more. ∎

4.3. *Failure of the Isomorphism Conjecture*

Can we say that $\leqslant_m^{1-L}$-complete degrees collapse completely? I.e., are the $\leqslant_m^{1-L}$ complete sets 1-L-isomorphic for the classes closed under log-lin reductions? The answer is no. It was shown in [6] that $\leqslant_m^{1-L}$-complete degree for PSPACE does not collapse to a single 1-L-isomorphic degree. We generalize this result for all classes closed under log-lin reductions. Our proof is simpler as well.

THEOREM 4.8. *For any class $\mathscr{C}$ closed under log-lin reductions, $\leqslant_m^{1-L}$-complete degree of $\mathscr{C}$, if it exists, does not collapse to a 1-L-isomorphic degree.*

Proof. Assume that for some class $\mathscr{C}$, the $\leqslant_m^{1-L}$-complete degree of $\mathscr{C}$ collapses to a 1-L-isomorphic degree. Let L be a $\leqslant_m^{1-L}$-complete set for $\mathscr{C}$. Let Xb and bX be the set of strings that are obtained by concatenating bit b, $b \in \Sigma$, at the end and beginning respectively, to each string of the set X. Define, $L_1 = L\mathbf{1}$ and $L_2 = \mathbf{0}L \cup \mathbf{1}\Sigma^*$. Sets L_1 and L_2 are clearly reducible to L via log-lin reductions, and therefore, both are in $\mathscr{C}$. It is also easy to see that both these sets are, in fact, $\leqslant_m^{1-L}$-complete for the class $\mathscr{C}$. Therefore, by our assumption, there is an isomorphism between L_1 and L_2 given by a 1-L function h, say. Since h is a reduction of L_1 to L_2, $h^{-1}(\mathbf{1}\Sigma^*) \subset L\mathbf{1}$. Since h is honest, there exists a polynomial p such that for every x, $p^{-1}(|x|) \leqslant |h(x)| \leqslant p(|x|)$. So, the set $h^{-1}(\mathbf{1}\Sigma^{=n})$ contains strings of at most $p(n)$ different sizes and since h is onto, there is an m, $m < p(n)$, such that $h^{-1}(\mathbf{1}\Sigma^{=n}) \cap \Sigma^{=m}\mathbf{1}$ contains at least $2^n/p(n)$ strings. Let $y\mathbf{1} \in h^{-1}(\mathbf{1}\Sigma^{=n}) \cap \Sigma^{=m}\mathbf{1}$ for such an m. Since h is a reduction of L_1 to L_2, $h(y\mathbf{0}) \in \mathbf{0}\Sigma^*$. Therefore, the 1-L TM M_h, computing h, before outputting any bit, must scan the whole input and check if the last bit is $\mathbf{0}$ or $\mathbf{1}$ for all such y's. Now there are $2^n/p(n)$ such y's. This means that when n is large enough so that $2^n/p(n)$ is greater than the total number of configurations of M_h of size $m+1$ (which is bounded by a polynomial in n), M_h will end up in the same configuration after reading first m bits of two different such input strings $y_1\mathbf{1}$ and $y_2\mathbf{1}$ of size $m+1$. M_h

will not have output anything by then and therefore its output on both the strings will be the same. This contradicts the fact that h is one-one. ∎

COROLLARY 4.9. *The* 1-*L-isomorphism conjecture is false.*

The above theorem, along with the Theorem 4.6, provides a tight upper and lower bound on the isomorphism of $\leqslant_m^{1-L}$-complete sets for almost all classes of interest.

5. 1-NL REDUCTIONS

The failure of the 1-L-isomorphism conjecture is due to the inability of 1-L TMs to carry out the Cantor–Schröder–Bernstein kind of construction of the isomorphism as in [11]. Moreover, Theorem 4.8 tells us that a second scan of the input cannot be avoided while computing the isomorphism. So now the question is—can one generalize 1-L reductions so that one can carry out the isomorphism construction within these reductions *and* the collapsing result still holds? That would provide us with an example of a reducibility for which the isomorphism conjecture is true. In this section, we answer this question affirmatively for 1-NL reductions.

The section is divided in two subsections. In the first subsection we prove a result similar to the Theorem 3.5 for 1-NL TMs, and in the next one we prove that $\leqslant_m^{1-NL}$ complete degrees collapse to 1-NL-isomorphic degrees for all classes closed under log-lin reductions

5.1. *Forgetful* 1-*NL TMs*

In this section, we shall only be concerned with 1-NL TMs computing functions. Configurations, and transit configurations, of 1-NL TMs are defined as for 1-L TMs. Since these TMs are nondeterministic, they can be at several configurations at any given time. So, the definition of a TM moving from a configuration to another is suitably altered:

DEFINITION 5.1. A 1-NL TM M *moves from configuration* C *to* D *on reading* s if $i(D) = i(C) + |s|$, D is a transit configuration, and the TM M when started on the configuration C with the string s written on the bit positions $i(C)$ through $i(D) - 1$ of the input, ends in the configuration D on one of its nondeterministic paths.

For a 1-NL TM M, let $config(M, n)$ be the set of configurations of size n as before. We also assume that $\|config(M, n)\| \leqslant q_M(n)$ for some polynomial q_M. We now define forgetful 1-NL TMs.

DEFINITION 5.2. A 1-NL TM is *forgetful* if for any n, there is a sequence of transit configurations of M of size n that is identical to the sequence of transit configurations of M on some accepting path for every input of size n. A function computed by a forgetful 1-NL TM is called a *forgetful* 1-NL function.

Note that in the above definition, we do not insist that the sequence of transit configurations is identical for all accepting paths on all inputs of size n. It suffices for our purpose as a 1-NL TM, by definition, must output the same strings on all accepting paths. We now prove the analog of the Theorem 3.5.

THEOREM 5.3. *For any class* $\mathscr{C}$ *closed under log-lin reductions, any* $\leqslant_m^{1-NL}$*-hard set for* $\mathscr{C}$ *is also hard under forgetful* 1-*NL reductions.*

Proof. Let A be a $\leqslant_m^{1-NL}$-hard set for C and B be an arbitrary set in $\mathscr{C}$, $B \neq 0, \Sigma^*$. We shall exhibit a reduction of B to A computed by a forgetful 1-NL TM. We first define a set D as accepted by the following procedure.

> Input y. Let $y = w\mathbf{01}^b\mathbf{0}^r$ for some $r \geqslant 0$. If $2b$ does not divide $|w|$ then reject.
> Otherwise, let $w = w_1 w_2 \cdots w_n$ where $|w_i| = 2b$ for $1 \leqslant i \leqslant n$. Define a new string x, $|x|$, such that $x[i] = \mathbf{1}$ if $w_i = uu$ for some string u, $\mathbf{0}$ otherwise. Accept iff $x \in B$.

Set D can clearly be reduced to B via a log-lin reduction and therefore, $D \in \mathscr{C}$. Let f be a 1-NL reduction of D to A computed by the TM M. We now define a reduction, g, of B to D based on the TM M. (Note that upto here we have just mimicked the proof of the Theorem 3.5, however, we cannot use the procedure to compute g as given there due to the presence of the nondeterminism). We first define the following four sets.

$$L_0 = \{(\mathbf{1}^m, \mathbf{1}^b, C, D) \mid C, D \in config(M, m), i(D) = i(C) + b + r + 1, \text{ and } D \text{ is an accepting configuration such that } M \text{ moves from } C \text{ to } D \text{ on reading } \mathbf{01}^b\mathbf{0}^r\},$$

$$L_1 = \{(\mathbf{1}^m, C, s, D) \mid C, D \in config(M, m), i(D) = i(C) + |s|, \text{ and } M \text{ moves from } C \text{ to } D \text{ on reading } s\},$$

$$L_2 = \{(\mathbf{1}^m, \mathbf{1}^b, C, D) \mid C, D \in config(M, m), i(D) = i(C) + 2b, \text{ and the number of strings } s \text{ of size } b \text{ such that } (\mathbf{1}^m, C, ss, D) \in L_1, \text{ is at least } 2 \cdot q_M(m)\},$$

$$L_3 = \{(\mathbf{1}^n, \mathbf{1}^m, \mathbf{1}^b, C) \mid C \in config(M, m), i(C) = 2bi + 1, \text{ and there exist } D_{i+1}, \ldots, D_{n+1} \text{ such that } (\mathbf{1}^m, \mathbf{1}^b, D_j, D_{j+1}) \in L_2 \text{ for } i \leqslant j < n \text{ (taking } D_i = C), \text{ and } (\mathbf{1}^m, \mathbf{1}^b, D_n, D_{n+1}) \in L_0\}.$$

It is easy to verify that all these sets are in NLOG. Now the following stage-wise procedure computes g.

Input x, with $|x| = n$.

Stage 0: Let $m = d_0 \cdot 2^{2\lceil \log n \rceil}$ and $b = \lceil 3 \cdot \log q_M(m) \rceil + \lceil \log n \rceil + 1$ (recall that $q_M(n)$ is the bound on the number of possible configurations of M of size n) where d_0 is the smallest number such that $m \geqslant (2n+1)\, b + 1$. Let C_0 be the initial configuration of M of size m, i.e., C^m_{init}.
Guess $\tilde{n}$ to be the length of the input such that $\lceil \log \tilde{n} \rceil = \lceil \log n \rceil$ (since the TM knows $\lceil \log n \rceil$, this can be done without scanning the input).

Stage j, $1 \leqslant j \leqslant \tilde{n}$: Find the smallest configuration C of M with $i(C) = 2bj + 1$, such that $(\mathbf{1}^m, \mathbf{1}^b, C_{j-1}, C) \in L_2$ and $(\mathbf{1}^{\tilde{n}}, \mathbf{1}^m, \mathbf{1}^b, C) \in L_3$. Now find the smallest two strings u and v, $u \neq v$ and $|u| = |v| = b$, such that $(\mathbf{1}^m, C_{j-1}, uu, C) \in L_1$ and $(\mathbf{1}^m, C_{j-1}, vu, C) \in L_1$. Let $C_j = C$ and $w_j = uu$ if $x[j] = 1$, vu otherwise.

Stage $\tilde{n} + 1$: At this stage, the TM can check if $\tilde{n} = n$, the correct length of the input. If it does not match, then abort. Otherwise, Let $r = m - (2n+1)b - 1$. Output the string $w_1 w_2 \cdots w_n \mathbf{01}^b \mathbf{0}^r$ and halt in an accepting state.

We now show that, on the path where $\tilde{n} = n$, the configuration C and the strings u and v, as required in the Stage j of the above procedure, can always be found. Suppose that $(\mathbf{1}^n, \mathbf{1}^m, \mathbf{1}^b, C_0) \in L_3$. Then by the definition of L_3, at every stage j, $1 \leqslant j \leqslant n$, a configuration C can be found such that $(\mathbf{1}^m, \mathbf{1}^b, C_{j-1}, C) \in L_2$ and $(\mathbf{1}^n, \mathbf{1}^m, \mathbf{1}^b, C) \in L_3$. By the definition of L_2 we know that there are at least $2 \cdot q_M(m)$ strings of the form ss with $|s| = b$, such that M moves from C_{j-1} to C on reading ss. Therefore, there must be at least $2 \cdot q_M(m)/q_M(m) = 2$ such strings, say uu and vv, such that M moves from C_{j-1} to some configuration D on reading either of u and v, and then moves from D to C on reading again either of u and v. Therefore, M moves from C_{j-1} to C on reading either of uu and vu as required.

So, all that we need to show now is that $(\mathbf{1}^n, \mathbf{1}^m, \mathbf{1}^b, C_0) \in L_3$. For $1 \leqslant i \leqslant n$, let X_i be the set of all sequences of transition configurations $D_0, D_1, ..., D_n, D_{n+1}, D_j \in config(M, m)$ for $0 \leqslant j \leqslant n+1$, such that (1) M moves from D_n to D_{n+1} on reading $\mathbf{01}^b\mathbf{0}^r$, and (2)$(\mathbf{1}^m, \mathbf{1}^b, D_i, D_{i+1}) \notin L_2$.

Now, let

$$S = \{ v_1 v_1 v_2 v_2 \cdots v_n v_n \mathbf{01}^b \mathbf{0}^r | (\forall i) |v_i| = b \}$$

where $r = m - (2n+1)b - 1$. Clearly, $\|S\| = 2^{nb}$. On how many strings in S can M have a transition configurations sequence belonging to the set X_i for some i? As there are less than $2 \cdot q_M(m)$ strings of the form ss such that M moves from D_i to D_{i+1} on reading ss for the pair D_i, D_{i+1} belonging to any sequence in X_i (by the definition of the set L_2), and there are at most $(q_M(m))^2$ possible choices for the pair D_i, D_{i+1}, it follows that less than $(q_M(m))^2 \cdot 2 \cdot q_M(m) \cdot 2^{(n-1)b}$ strings in S have a transition configurations sequence belonging to the set X_i. Therefore, less than $2 \cdot n \cdot (q_M(m))^3 \cdot 2^{(n-1)b} \leqslant 2^{nb}$ strings in S have a transition configurations sequence belonging to the set $\bigcup_{1 \leqslant i \leqslant n} X_i$. This implies that there is at least one string in S such that the transition configurations sequence of M on some accepting path for this string does not belong to the set $\bigcup_{1 \leqslant i \leqslant n} X_i$. The existence of such a sequence proves that $(\mathbf{1}^n, \mathbf{1}^m, \mathbf{1}^b, C_0) \in L_3$.

The entire procedure above can be carried out by a logspace NTM since the length of the configurations and b is bounded by $O(\log n)$, and since NLOG is closed under complement [14, 22]. The input head of the TM needs only be one-way as it needs to scan the input x only once from left to right. Therefore, the function g is a 1-NL function. It is clear, from the definition of the set D that g is a reduction of B to D. Therefore, $h = f \circ g$ is a reduction of B to A. We now show that h can be computed by the following forgetful 1-NL TM.

On input x, $|x| = n$, the TM first computes $\mathbf{1}^{\lceil \log |g(x)| \rceil} = \mathbf{1}^{\lceil \log d_0 \rceil + 2\lceil \log n \rceil}$ using the string $\mathbf{1}^{\lceil \log n \rceil}$ written on the worktape. Now it simulates the TM M on $g(x)$ while computing $g(x)$ according to the above procedure in parallel. The TM has, initially, the initial configuration of M on the input $g(x)$, say C_0, written on the worktape. For every j, $1 \leqslant j \leqslant n$, the TM, before reading the jth bit of the input, takes the configuration C_{j-1} of M and, using the Stage j of the procedure above, computes the two strings u and v and the configuration C_j. Now it scans the jth input bit and computes the string w_j with $w_j = uu$ if $x[j] = \mathbf{1}$, vu otherwise. Then, it continues with the simulation of M on w_j. At the end of this simulation, M will end up (in one of its accepting paths) in the configuration C_j no matter which of the two values w_j takes. After scanning all bits of the input, the TM continues the simulation of M on the string $\mathbf{01}^b\mathbf{0}^r$ where $r = m - (2n+1)b - 1$ as above.

It is easy to see that the above TM is a forgetful 1-NL TM computing the function h. ∎

5.2. *Collapse of Complete Degrees*

In this subsection we will prove that for every class closed under log-lin reduction, 1-NL isomorphism conjecture is true. Towards this we first prove a collapse result similar to Theorem 4.2 for $\leqslant_m^{1-L}$ complete degrees.

THEOREM 5.4. *For any class $\mathscr{C}$ closed under log-lin reductions, every $\leqslant_m^{1-NL}$-hard set for $\mathscr{C}$ is also $\leqslant_{1,\,qli,\,i}^{1-NL}$-hard.*

Proof. Let A be a $\leqslant_m^{1-NL}$-hard set for $\mathscr{C}$. Let $B \in \mathscr{C}$, $B \neq \Sigma^*, \varnothing$. We construct a reduction h' of B to A exactly as

in the proof of the Theorem 4.2. So, $h' = f' \circ g'$ with $|g'(x)| = 2^{2r(l)}$ for a suitable l. By exactly the same arguments as there, it can be shown that h' is a one-one and length-squaring forgetful 1-NL function. We now show that h' is also 1-NL-invertible.

As in the proof of the Theorem 4.2, we define the 1-NL TM computing h'^{-1} as a composition of two 1-NL TMs M_1 and M_2 where M_1 computes f'^{-1} on the range of h' and M_2 computes g'^{-1}. Let f' be computed by the forgetful 1-NL TM M'. The following three sets would be useful in the computation of M_1.

$L_0 = \{(\mathbf{1}^n, a, C, D) | C, D \in config(M, n), D$ is a transit configuration, $i(D) = i(C) + 1$, and M' moves from C to D on reading the bit $a\}$,

$L_1 = \{(\mathbf{1}^n, C, D) \mid C, D \in config(M', n)$ are transition configurations, $i(D) = i(C) + 1$, M' moves from C to D on reading either of the bits $\mathbf{0}$ and $\mathbf{1}\}$,

$L_2 = \{(\mathbf{1}^n, C, D) \mid C, D \in config(M', n), D$ is a transit configuration with, $i(C) = j + 1$, and there exist transit configurations $D_{j+1}, ..., D_n$ such that D_n is an accepting configuration and for every $j \leqslant i \leqslant n$, $(\mathbf{1}^n, \mathbf{d}_i, D_{i+1}) \in L_1$ (taking $D_j = C)\}$.

It is easy to see that all these sets are in NLOG. The TM M_1 carries out the following procedure.

Input z. Compute an l such that, letting $n = 2^{2r(l)}$, if $w = f'(\mathbf{1}^n)$ then $\mathbf{1}^{\lceil \log |w| \rceil} = \mathbf{1}^{\lceil \log |z| \rceil}$. There will be *at most* one such l since $|h'(x)| \leqslant 1/2 \cdot |h'(y)|$ if $|g'(x)| < |g'(y)|$ (as argued in the proof of the Theorem 4.2). If there is no such l then halt in a rejecting state. Otherwise, do the following stage-wise computation (if $h'^{-1}(z)$ is defined then $|f'^{-1}(z)|$ must be n).

Stage 0: Let $C_0 \in config(M', n)$ be the initial configuration. Since M' is forgetful, we are guaranteed that $(\mathbf{1}^n, C_0) \in L_2$.

Stage j, $1 \leqslant j \leqslant n$: Find the smallest configuration C_j such that $(\mathbf{1}^n, C_{j-1}, C_j) \in L_1$ and $(\mathbf{1}^n, C_j) \in L_2$. There must be such a configuration since M' is forgetful. Now we compute the outputs of M' while moving from C_{j-1} to C_j on reading the bits $\mathbf{1}$ and $\mathbf{0}$. Let $C = C_{j-1}$, and $o_1 = \varepsilon$. Repeat the following till $C = C_j$: Find a configuration C' such that M' moves from C to C' in a single step on reading the bit $\mathbf{1}$, and $(\mathbf{1}^n, \mathbf{1}, C', C_j) \in L_0$; append to o_1 any output produced by M' in this step; let $C = C'$ and continue. Similarly compute o_0 the output of M' on reading bit $\mathbf{0}$. Since M' is forgetful and f' is one-one on strings of equal length, $|o_1| = |o_0|$ and $o_1 \neq o_0$. Check which of these is a prefix of the string that is to the right of the input head. If none is, then halt in a rejecting state. Otherwise, output the bit whose output is a prefix and goto next stage.

Stage $n + 1$: Halt in an accepting state.

Using the facts that M' outputs the same string on all its accepting paths on any input and that NLOG is closed under complement [14, 22], it is easy to see that M_1 works correctly and is a 1-NL TM. Moreover—and this would be important in the context of computing the isomorphism—M_1 is a *strong* NTM, i.e., on any input, it never both accepts and rejects on two different paths.

The TM M_2 computing the inverse of g' is identical to the one described in the proof of the Theorem 4.2. It is actually a 1-L TM. The TM computing the inverse of h' would simulate M_1 and M_2 in parallel. It would also be a strong NTM as M_1 is and M_2 is deterministic. ▮

COROLLARY 5.5. *For any class $\mathscr{C}$ closed under log-lin reductions, $\leqslant_m^{1-NL}$-complete degree of $\mathscr{C}$ collapses to a single 1-NL-isomorphic degree.*

Proof Sketch. Take any two sets A and B that are $\leqslant_m^{1-NL}$-complete for $\mathscr{C}$. By Theorem 5.4, we have that they are reducible to each other via one-one, length-squaring and 1-NL-invertible reductions. To construct the isomorphism, we use the same construction as in the proof of Theorem 4.6. The function t is defined as before, with functions u and v being one-one, length-squaring, 1-NL-invertible, 1-NL reductions between A and B. Function t can be computed by a non-deterministic logspace TM with two scans of the input—first to compute the length of the inverse chain and seconds to output. To compute the length of the inverse chain, the same procedure as given in the proof of the Theorem 4.6 is followed. The crucial point to note here is that the 1-NL TMs computing the inverses of u and v are strong NTMs. So, whenever, on some intermediate string, the inverse is not defined, the corresponding TM ends up in a rejecting state on some path and this can easily be detected by the procedure.

Using non-determinism, one can combine these two scans in one in the following way. At the beginning of the computation on x, *guess* the length of the chain. Now, verify the guess while simultaneously outputting $u(x)$ or $v^{-1}(x)$ based on the guessed length. If a guess turns out to be wrong, abort the computation on that path. Thus, in a single scan, one gets the output. Similarly, t^{-1} can also be computed. ▮

COROLLARY 5.6. *The 1-NL-isomorphism conjecture is true.*

This is the first non-trivial example of a reducibility for which the isomorphism conjecture holds.

6. 1-omL REDUCTIONS

In the previous two sections, we have seen examples of reducibilities for which the encrypted complete set conjecture fails while the isomorphism conjecture fails for one and holds for the other. In this and the next section, we study two reducibilities, viz., 1-omL and c-L, such that the encrypted complete set conjecture fails for one and holds for the other. These two reducibilities also exhibit an interesting property—the complete degrees under both are the *same* even though the class of 1-omL functions is larger than the class of c-L functions. There is another reason for investigating the complete degree under 1-omL reductions—these reductions are powerful enough to possibly include *p-one-way* functions. We first identify such functions.

Recall that p-one-way functions exist if and only if $\mathrm{P} \neq \mathrm{UP}$ [16, 10]. Assuming $\mathrm{P} \neq \mathrm{UP}$, one can construct the following "natural" class of p-one-way functions.

Take any set $A \in \mathrm{UP} - \mathrm{P}$. Let M be a polynomial-time NDTM accepting A. A typical encoding of an accepting computation of M on input x is of the form $ID_1 ID_2 \cdots ID_m$ where $m = p(|x|)$ for some polynomial p, $|ID_i| = m$, ID_1 and ID_m are the starting and accepting IDs respectively of M, and for every i, $1 \leqslant i \leqslant m$, M moves from ID_i to ID_{i+1} in a single step. Define the function f_M as: $f_M(y) = \mathbf{1}x$ if y is the above encoding of the accepting computation of M on x; $\mathbf{0}y$ otherwise. Now we have,

PROPOSITION 6.1. *Function f_M is a 1-omL function, and if $P \neq UP$, then f_M^{-1} is not computable in polynomial-time.*

Proof Sketch. It is easy to see that f_M is not p-invertible. An accepting computation y of M on input x is of the form $ID_1 ID_2 \cdots ID_m$, $m = p(|x|)$, and each ID_i itself is of length m. The function f_M can be computed by the following 1-omL TM with four input heads—on input y, calculate the length of the input using the first head, check if it is equal to $[p(n)]^2$ for some n. If not then output $\mathbf{0}y$. Otherwise, compare successive IDs using the next two heads to ensure that they form a valid computation, the first ID is the initial ID and the final ID is an accepting one; if it is indeed an accepting computation then using the last head extract x and output $\mathbf{1}x$ else output $\mathbf{0}y$. In all these calculations, obliviousness of the input heads can be easily maintained. ∎

We now show that the complete degree under 1-omL reductions is the same as the one under c-L reductions. The collapse result for c-L reductions will be proven in the next section. For a 1-omL TM, we can assume, without loss of generality, that the jth input head of the TM is never ahead of the ith one, for $j > i$. A configuration of a 1-omL TM stores the position of all the heads along with the contents of the work tape and the state of the TM. We let, as usual, $config(M, n)$ denote the set of configurations of the 1-omL TM M on input strings of size n, and $q_M(n)$ the polynomial bounding the number of configurations in $config(M, n)$.

THEOREM 6.2. *For any class $\mathscr{C}$ closed under log-lin reductions, any $\leqslant_m^{1-omL}$-hard set for $\mathscr{C}$ is also $\leqslant_m^{c-L}$-hard.*

Proof. Let A be a $\leqslant_m^{1-omL}$-hard set for $\mathscr{C}$ and B be an arbitrary set in $\mathscr{C}, B \neq \Sigma^*, \varnothing$. Define a set D as accepted by the following procedure.

> Input y. If $|y|$ is not even, then reject. Otherwise, let $y = ws$ with $|w| = |s| = |y|/2$. Let $j_1, j_2, ..., j_t$, $1 \leqslant j_1 < j_2 < \cdots j_t \leqslant |s|$, be the sequence of all bit positions where the string s is **1**. Define string x as $x = w[j_1]\, w[j_2] \cdots w[j_t]$. Now, accept iff $x \in B$.

It is easy to see that $D \leqslant_m^{log\text{-}lin} B$ and so, $D \in \mathscr{C}$. Let f be a 1-omL reduction of D to A computed by the TM M having a input heads. We define a reduction, g, of B to D, based on the TM M, as given by the stage-wise procedure below. Function g, on input x, outputs a string on which the TM M can be simulated by a c-L TM. String $g(x)$ has, at some chosen bit positions the bits of x written, and the rest of the bits of the string depend only on $|x|$ and not on x. We shall refer to the bits of $g(x)$ where bits of x are written as *live* bits and the rest of the bits as *dead* bits. The heads of M, on input $g(x)$, can be divided into d groups, $d \leqslant a$, such that while the heads in the kth group, $1 \leqslant k \leqslant d$, have any live bits to scan, the heads in the $(k+1)$th group (if $k < d$) have not scanned any live bits. Further, for any live bit, all the heads in the kth group scan it before any of the heads in the group scans the next live bit. This allows the TM to be simulated by a c-L TM that makes d scans of the input $g(x)$ simulating the scan of live bits by the kth group of heads of M in its kth scan of the input (the TM can find the value of dead bits by computing $g(\mathbf{1}^{|x|})$).

> **Input** x, with $|x| = n$.
>
> **Stage** 0: Make a scan of the input to compute n. Let $c_b = 1$, and $c_g = 1$. Let $r(k) = 1/2 \cdot (2n)^{2^{k-1}}$, $H_1 = \{1\}$ and $H_k = \varnothing$ for $1 < k \leqslant a$ (the set H_k will eventually have the kth group of heads as described above). Let $v = \mathbf{1}^{2r(a)}$ and assume every bit of v to be live. (The procedure keeps the track of live bits through the counters c_b and c_g.)
>
> /* The following invariant will hold at the beginning of every stage j, $j \geqslant 1$: the live bits of v are $v[c_b + i \cdot c_g]$ for $0 \leqslant i \leqslant r(a-j+1) - 1$.*/
>
> **Stage** j, $1 \leqslant j < a$: Let $j \in H_k$. Simulate the TM M on the input v. Divide the first $r(a-j) \cdot (r(a-j)+1)$ live bits of v in $r(a-j)$ groups of $1 + r(a-j)$ successive live bits each. Since $r(a-j) \cdot (r(a-j)+1) \leqslant r(a-j+1)$, such a division is possible. Check if, during the simulation, for every group of live bits, by the time the

jth head scans all the bits in the group, the $(j+1)$th head scans the first bit in the group. If there is a group, say the lth group, for which this is not true, then retain only the bits in the lth group, except for the first one, as live bits, and assume all the other bits to be dead. Set c_b to the position of the first live bit of the lth group, i.e., $c_b = c_b + (l-1)\cdot(r(a-j)+1)\cdot c_g + c_g$, and let $H_{k+1} = \{j+1\}$. Otherwise, if for every group the condition holds, then retain only the first bit of each group as live bit, and assume all the other bits to be dead. Set c_g to the distance between live bits, i.e., $c_g = c_g \cdot (r(a-j)+1)$, and let $H_k = H_k \cup \{j+1\}$.

Stage a: (At the beginning of this stage, there are $r(1) = n$ live bits.) Compute w and s, $|w| = |s| = r(a)$, such that $w[m] = x[i+1]$ and $s[m] = \mathbf{1}$ if $m = c_b + i \cdot c_g$ for some $0 \leqslant i < n$, otherwise $w[m] = s[m] = \mathbf{0}$. Output ws.

Function g, as computed by the above procedure is clearly a reduction of B to D. It is also clear by the construction, that the heads of the TM M, on input $\mathbf{1}^{2r(a)}$, can be divided into d groups $H_1, H_2, ..., H_d$, for some $d \leqslant a$, such that the above described property holds. Since $|g(x)| = 2r(a)$ and the TM M is oblivious, its input heads move in an identical manner on input $g(x)$, and therefore, the same property holds for M on input $g(x)$ too. A 2-L TM can compute $g(x)$ since the numbers c_b and c_g are polynomially bounded, the simulation of M on $v = \mathbf{1}^{2r(a)}$ can be carried out in logspace, and the TM needs to scan the input only twice (the first scan to compute $|x|$ and the second to compute w).

Let $h = f \circ g$. Function h is a reduction of B to A, and it can be computed by a c-L TM that, on input x, in parallel, computes $g(x)$ and simulates M on it. ∎

The above result immediately raises the following question: is the class of c-L functions properly contained in the class of 1-omL functions? If so, then we have an interesting collapse of $\leqslant_m^{1-omL}$-complete degrees to complete degrees under a strictly weaker class of reductions. The following proposition shows that it is indeed so.

PROPOSITON 6.3. $\mathscr{F}(c\text{-}L) \subset \mathscr{F}(1\text{-}omL)$.

Proof Sketch. Any c-L function f can be computed by an *oblivious* c-L TM in the following way—suppose that a TM, say M, computing f, works for at most $p(n)$ steps on input strings of size n. Then a TM that takes exactly $p(n)$ steps to simulate steps of M between two consecutive movements of the input head will be an oblivious c-L TM computing f since the movement of the input head is one-way. Now, this oblivious TM can be simulated by a 1-omL TM—put one head for each scan of the input. Therefore, $\mathscr{F}(\text{c-L}) \subseteq f(\text{1-omL})$.

By the Proposition 7.10 proved below, the inverse of the function $t(x) = xx$ is computable by a 1-omL TM but not by a c-L TM. This completes the proof. ∎

7. c-L REDUCTIONS

The section is divided in two subsections. In the first one, we show that the complete sets under c-L reductions are also complete under forgetful c-L reductions, and in the second one we show the collapse result for these reductions as well as prove the c-L-encrypted complete set conjecture.

7.1. *Forgetful c-L TMs*

All the notions for 1-L TMs in section 3 can be defined analogously for c-L TMs. We now show that,

THEOREM 7.1. *For any class $\mathscr{C}$ closed under log-lin reductions, any $\leqslant_m^{c-L}$-hard set for $\mathscr{C}$ is also hard under forgetful c-L reductions.*

Proof. Let A be a $\leqslant_m^{c-L}$-hard set for $\mathscr{C}$ and $B \in \mathscr{C}$, $B \neq \varnothing, \Sigma^*$. We shall exhibit a reduction of B to A computed by a forgetful c-L TM. We first define a set D—exactly as in the proof of the Theorem 3.5—to be accepted by the following procedure.

Input y. Let $y = w\mathbf{01}^b\mathbf{0}^r$ for some $r \geqslant 0$. If $2b$ does not divide $|w|$ then reject. Otherwise, let $w = w_1 w_2 \cdots w_n$ where $|w_i| = 2b$ for $1 \leqslant i \leqslant n$. Define a string x, $|x| = n$, such that $x[i] = \mathbf{1}$ if $w_i = uu$ for some string u, $\mathbf{0}$ otherwise. Accept iff $x \in B$.

For the set D, we have that $D \in \mathscr{C}$ and therefore, $D \leqslant_m^{c-L} A$ via f computed by the TM M. Again as in the proof of the Theorem 3.5, we define a reduction, g, of B to D, based on the TM M. However, since the TM M makes more than one scan of the input tape, the construction of g would be different. Let M make a scans of the input tape on any input. Function g is computed by the following stage-wise procedure.

Input x, with $|x| = n$.

Stage 0: Let $m = d_0 \cdot n^2$ and $b = \lceil 2a \cdot \log q_M(m) \rceil + 1$ (recall that $q_M(n)$ is the bound on the number of possible configurations of M of size n) where d_0 is the smallest number such that $m \geqslant (2n+1)\,b + 1$. Let C_0^1 be the initial configuration of M of size m, i.e., C_{init}^m.

Stage $n(k-1)+j$, $1 \leqslant k \leqslant a$, $1 \leqslant j \leqslant n$: (The configurations C_{j-1}^i, $1 \leqslant i \leqslant k$ are computed in the previous stage.) If $k > 1$, then compute C_j^i, D_j^i, $1 \leqslant i < k$, from the configurations C_{j-1}^i, $1 \leqslant i < k$, as in the Stage $n(k-2)+j$. Now, find the smallest configurations D_j^k and C_j^k, $i(D_j^k) = 2bj - b + 1$ and $i(C_j^k) = 2bj + 1$, satisfying the following condition:

there are at least $2 \cdot (q_M(m))^{2 \cdot (a-k)}$ different strings of size b each, such that for every i, $1 \leqslant i \leqslant k$, the TM M moves from C^i_{j-1} to D^i_j *and* from D^i_j to C^i_j on reading any of these strings.
If $k = a$, then find the smallest two strings of length b, say u and v, that satisfy the above condition. Let $w_j = uu$ if $x[j] = \mathbf{1}$, vu otherwise.
If $j = n$ and $k < a$ (the input head is at the end of the tape and there are more scans to be made), then, for each i, $1 \leqslant i \leqslant k$, compute the configurations C^{i+1}_0.

Stage $nd+1$: Let $r = m-(2n+1)b-1$. Output the string $w_1 w_2 \cdots w_n \mathbf{01}^b\mathbf{0}^r$.

We now show that the configurations C^i_j and D^i_j as required in the Stage $n(k-1)+j$ of the above procedure, can always be found. The proof is by induction on k. Base step ($k=1$): there must exist a configuration D^1_j such that there are at least $2^b/q_M(m)$ strings of size b reading any of which M moves from C^1_{j-1} to D^i_j. Now, amongst these $2^b/q_M(m)$ strings, there are at least $2^b/(q_M(m))^2$ strings and a configuration C^1_j such that M moves from D^1_j from C^1_j on reading any of these strings. So, there are at least $2^b/(q_M(m))^2 \geqslant 2 \cdot (q_M(m))^{2 \cdot (a-1)}$ strings on reading any of which the TM M moves from C^1_{j-1} to D^1_j and from D^1_j to C^1_j. The argument for the induction step is identical. When $k = a$, the required strings u and v exist since $2 \cdot 2^{2 \cdot (a-a)} = 2$.

The above procedure is a recursive one: the Stage $n(k-1)+j$ executes the Stage $n(k-2)+j$ for $k > 1$. It can be computed by a logspace bounded DTM, say M_g, since the depth of the recursion is bounded by a—a constant, and the length of the configurations and b is bounded by $O(\log n)$. Also, in the stage $n(k-1)+n$ for $k < a$, the configuration C^{i+1}_0 can be easily computed from the configuration C^i_n since no input bit is scanned (only the input head is moved from the end of the input to the beginning). The TM M_g requires only a single, left to right, scan of the input to compute the strings w_j. As the value of w_js depend only on the jth input bit, M_g need only be forgetful. Therefore, g is a forgetful 1-L function. It is also clear, from the definition of the set D that g is a reduction of B to D.

Let $h = f \circ g$. Function h is a c-L reduction of B to A. It can be computed by a forgetful c-L TM, say M_h, that, on input x, first computes the length n of the input and from it $|g(x)| = m = d_0 \cdot n^2$. Now it simulates the TM M on $g(x)$ and M_g on x in parallel. Since M_g is forgetful, after scanning the jth bit of the input, M_g would end up in a configuration that is independent of x. The output of M_g would be w_j during the scan of this bit, and w_j is constructed such that the TM M would end up, on scanning w_j, and during any of its a scans, in a configuration that is independent of the two possible values of w_j. Therefore, the TM M_h is forgetful. ▌

7.2. *Collapse of Complete Degrees*

In a fashion similar to the Theorems 4.2 and 5.4, we show that

THEOREM 7.2. *For any class $\mathscr{C}$ closed under log-lin reductions, any $\leqslant^{c-L}_m$-hard set for $\mathscr{C}$ is also $\leqslant^{c-L}_{1,\,qli}$-hard under reductions that are* 1*-omL-invertible.*

Proof. Let A be a $\leqslant^{c-L}_m$-hard set for $\mathscr{C}$. Let $B \in \mathscr{C}$, $B \neq \Sigma^*, \varnothing$. We construct, again, a reduction h' of B to A exactly as in the Stage 2 of the proof of the Theorem 4.2. So, $h' = f' \circ g'$ with $|g'(x)| = 2^{2r(l)}$ for a suitable l. By exactly the same arguments as there, it can be shown that h' is a one-one, length-squaring, forgetful c-L function. We now show that h' is also 1-omL-invertible. Let h' be computable by the forgetful c-L TM M_h that makes a scans of the input. The following procedure computes the inverse of h'.

Input z. Compute the length m of the input. Now, check if there is a number n such that $|h'(\mathbf{1}^n)| = m$. Reject if not. Otherwise proceed as follows (if the inverse is defined, then its length must be n since h' is forgetful implying that $|h'(y)|$ is the same for every string y of length n). Simulate again M_h on $\mathbf{1}^n$ and calculate the number of bits output by it during its ith scan of the input for $1 \leqslant i \leqslant a$ (by the same property as above this number must be the same for the actual inverse, if it exists). Let this number be o_i for $1 \leqslant i \leqslant a$. Place one input head each on the input bits $z[o_i+1]$ for $0 \leqslant i < a$ ($o_0 = 0$). Simulate M_h to produce its output on the first bit of its input during all the scans and for both possible values of the bit. Compare the outputs with the strings written to the right of the input heads. If the outputs for neither bit values match, then halt in a rejecting state. Otherwise, output the bit whose output matches (the outputs on the two bits must be of the same length and different on at least one scan since M_h is forgetful and h' is one-one). Repeat the same process for the rest of the $n-1$ bits of the input to M_h. If the entire input matches with the output of M_h on some string, then halt in an accepting state, otherwise halt in a rejecting state.

It is easy to see that the above procedure can be carried out by a 1-omL TM. ▌

COROLLARY 7.3. *For any class $\mathscr{C}$ closed under log-lin reductions, the $\leqslant^{c-L}_m$-complete degree for $\mathscr{C}$ collapses to the $\leqslant^{c-L}_{1,\,qli}$-complete degree.*

As the $\leqslant^{1-omL}_m$-complete degrees are the same as $\leqslant^{c-L}_m$-complete degrees (Theorem 6.2), we have

COROLLARY 7.4. *For any class $\mathscr{C}$ closed under log-lin reductions, the $\leqslant^{1-omL}_m$-complete degree of collapses to the $\leqslant^{1-omL}_{1,\,qli,\,i}$-complete degree.*

COROLLARY 7.5. *For any class $\mathscr{C}$ closed under log-lin reductions, if A is a $\leqslant_m^{1-omL}$-hard set for $\mathscr{C}$ and t is a one-one, 1-omL function then $t(A)$ is also $\leqslant_m^{1-omL}$-hard.*

Proof. We first observe that the function h' of the above theorem is a forgetful c-L function as it is a composition of two forgetful c-L functions. It is easy to see that a composition of a 1-omL function with a forgetful c-L function remains a 1-omL function. Therefore, $t \circ h'$ is a 1-omL function implying that $t(A)$ is 1-omL-hard. ▌

The above corollary along with the Theorems 6.2 and 7.2 imply that,

COROLLARY 7.6. *The 1-omL-encrypted complete set conjecture is false.*

COROLLARY 7.7. *For any class $\mathscr{C}$ closed under log-lin reductions, the $\leqslant_m^{1-omL}$-eomplete sets for $\mathscr{C}$ are logspace-isomorphic.*

Is the 1-omL-isomorphism conjecture true? We suspect not, as to compute the length of $f^{-1} - g^{-1}$ chain as in [11], it appears that a polynomial number of input heads are needed instead of just a constant. We have not been able to prove it though. For c-L reductions, however, we prove that the c-L-encrypted complete set conjecture is true and therefore, the c-L-isomorphism conjecture is false. Towards this, we first define c-L-annihilating functions in the same spirit as [18].

DEFINITION 7.8. A function f is a *c-L-annihilating* function if it is a one-one, length-increasing, c-L function such that every subset of the range of f that is recognized by a c-L TM is sparse.

PROPOSITION 7.9. *If c-L-annihilating functions exist then the c-L-encrypted complete set conjecture is true.*

Proof. Suppose the conjecture is false. Let A be a $\leqslant_m^{c-L}$-complete set for NP and f be a c-L-annihilating function. Define $B = \mathbf{1}A \cup \Sigma^*$. Set B too is $\leqslant_m^{c-L}$-complete for NP. Consider the set $f(B)$. Since the conjecture is false, there is a one-one, length-increasing, c-L function g reducing B to $f(B)$ such that g is c-L-invertible as well. Define the set C as: $x \in C$ iff $g^{-1}(x)$ is defined and belongs to $\mathbf{0}\Sigma^*$. C is a non-sparse set recognizable by a c-L TM as well as a subset of the range of f. A contradiction. ▌

It is easy to show that c-L-annihilating functions exist. Define $t(x) = xx$.

PROPOSITION 7.10. *t is a c-L-annihilating function.*

Proof. Function t is clearly a one-one, length-increasing, c-L function. Let the c-L TM M recognize a subset S of its range. Let M be a k-L TM with $q_M(2n)$ being the number of configurations in *config*$(M, 2n)$. We show that $\|S_{=2n}\| \leqslant [q_M(2n)]^{3k}$. Consider the sequences of transit configurations (of size $2n$) $C_1^1, C_2^1, C_3^1, C_1^2, C_2^2, C_3^2, \ldots C_1^k, C_2^k, C_3^k$, where $i(C_1^j) = 1$, $i(C_2^j) = n+1$, and $i(C_3^j) = 2n+1$ respectively for $1 \leqslant j \leqslant k$. How many different such sequences exist? Clearly, not more than $(q_M(2n))^{3k}$ as there are $3k$ configurations in any such sequence. Suppose that there are two strings in S, say xx and yy, $x \neq y$, that share the same sequence. Then the strings xy and yx would also have the same sequence, and therefore, they would also belong to S. But this is not possible since $x \neq y$ and S is a subset of the range of t. Therefore, S can contain at most $(q_M(2n))^{3k}$ strings of size $2n$. Note that S has no string of odd length. Therefore, S is sparse. Since the TM M was arbitrary, it follows that t is a c-L-annihilating function. ▌

COROLLARY 7.11. *c-L-encrypted complete set conjecture is true.*

8. DISCUSSION

The motivation for relocating the conjectures to weaker reducibilities, or to higher classes, has been that an answer of the relocated conjectures may shed some light on the answer of the conjectures in their original form. So, what, if any, is the implication of the above results in this sense? At a first glance, they do not seem to favor any conjecture as all the three possible answers to the two conjectures have been shown to exist for different reducibilities—both are false for 1-L while the isomorphism conjecture is true for 1-NL and the encrypted complete set conjecture is true for c-L reductions. However, on a closer look these results appear to support the p-isomorphism conjecture. To see this, we first identify two properties of a reducibility r.

DEFINITION 8.1. Reducibility r is *simple* for a class $\mathscr{C}$ if every $\leqslant_m^r$-complete set for $\mathscr{C}$ is also $\leqslant_{1,li}^r$-complete.

DEFINITION 8.2. Reducibility r is *deterministically invertible* for a class $\mathscr{C}$ if for every resource bound s, $s \geqslant r$—given that the inverse of every one-one, length-increasing function in $\mathscr{F}(r)$ is computable by a *non-deterministic* TM working within the resource bound of s—any $\leqslant_{1,li}^r$-complete set for $\mathscr{C}$ is also $\leqslant_{1,li}^r$-complete via reductions whose inverses are computable by *deterministic* TMs working within the resource bound of s.

The following conjecture asserts that both of the above properties hold for a class $\mathscr{C}$ and reducibility r.

THE r-COMPLETE-DEGREE CONJECTURE FOR $\mathscr{C}$. *Reducibility r is both simple and deterministically invertible for the class $\mathscr{C}$.*

It is straightforward to see that the p-isomorphism conjecture holds if and only if the p-complete degree conjecture holds for NP as the inverse of any one-one, length-increasing polynomial-time function is computable by a NTM working in polynomial-time. We now show that

for the reducibilities 1-L, 1-omL, and c-L—for which we could not prove the isomorphism conjecture—the *r*-complete degree conjecture for NP either holds or is likely to hold.

That the conjecture holds for the reducibilities 1-L and 1-omL follows directly from the Theorem 4.2 and the Corollary 7.4 respectively. In fact, as shown in the Proposition 6.1, there are 1-omL functions whose inverses are computable only by polynomial-time NTMs and yet the $\leqslant_{1,li}^{1-omL}$-complete sets for NP are also $\leqslant_{1,li,i}^{1-omL}$-complete. So, for 1-omL reductions, the inverses are stronger than desired.

The interesting case is that of c-L reductions. The c-L reducibility is simple by the Theorem 7.2. Is it also deterministically invertible? It can be shown—by a direct adaptation of the proof of the Proposition 7.10—that there are one-one, length-increasing c-L functions whose inverses are *not* computable by non-deterministic c-L TMs ($t(x) = xx$ is one such function). In fact, there seems no better way to invert an arbitrary one-one, length-increasing c-L function than by a non-deterministic 1-mL TM—keep one head for the output produced during each scan of the c-L TM; guess the length of the output produced during each scan and position the heads accordingly; now guess the input to the c-L TM bit-by-bit and verify its output. Therefore —as by the Theorem 7.2, the $\leqslant_{1,li}^{1-omL}$-complete sets for NP are also $\leqslant_{1,li}^{1-omL}$-complete via reductions that are 1-omL-invertible—the c-L reducibility too *appears* to be deterministically invertible for NP. Note that it is hard to prove that the c-L reducibility *is* deterministically invertible for NP as that would require us to show the determinising property for *every* resource bound *s*, $s >$ c-L, using which a NTM can compute the inverse of one-one, size-increasing c-L functions.

The 1-NL-complete degree conjecture is clearly true (follows from the Theorem 5.4). The only point to note is that in the definition of the deterministically invertible property we require the resource bound on the inverting TM to be at least as much as *r* which in case of 1-NL TMs is 1-NL (we count non-determinism also as a resource). And so the conclusion that a deterministic TM must compute the inverse within a resource bound of *s* is satisfied as non-determinism is still allowed in the resource bound *s*.

Thus, for all the four reducibilities that we have considered, the *r*-complete degree conjecture for NP (in fact for any class closed under log-lin reductions) is true (or likely to be true). So, our results can be interpreted as providing evidence for the *p*-isomorphism conjecture provided one believes that the complete degrees under these weak reducibilities have a similar "structure" as the complete degree under polynomial-time reducibility. However, this is far from clear. The reducibilities that we considered have the special property that the complete sets under them are also complete under forgetful reductions—a very weak kind of reductions. And in the proof of our collapse results, we make heavy use of this property. On the other hand, for polynomial-time (even logspace) reductions, it is easy to show that there are $\leqslant_m^p$-complete sets which are not complete under reductions computed by forgetful TMs with arbitrary resources.

In view of the above, more investigation is needed—particularly of the *r*-complete degree conjectures for various other reducibilities—before we can arrive at some conclusion regarding the isomorphism conjecture.

Finally, a few words on the technique. The technique that we use is essentially a refinement of the one used in [1]. Our results show that this technique is more useful than the standard diagonalization one at least for weak reductions like 1-L, 1-NL etc. For example, in [13], it was shown, using the diagonalization technique, that $\leqslant_m^{1-L}$- (and $\leqslant_m^{1-NL}$-) complete sets for the nondeterministic space classes above NLOG are also $\leqslant_{1,li}^{1-L}$-(resp. $\leqslant_{1,li}^{1-NL}$-) complete. We have been able to improve this result in two ways—one, we show that the $\leqslant_m^{1-L}$- and $\leqslant_m^{1-NL}$-complete sets are also respectively $\leqslant_{1,li,i}^{1-L}$-and $\leqslant_{1,li,i}^{1-NL}$-complete; and two, that this result holds for all classes closed under log-lin reductions.

9. OPEN QUESTIONS

As we have observed above, it would be interesting to investigate the *r*-complete degree conjecture for NP for various reducibilities *r*, the most important ones being, of course, logspace and polynomial-time. We list here a couple of reducibilities for which the answer to the conjecture appears tractable.

1-*mL Reductions.* We have not been able to answer the conjecture for the more natural class of 1-mL reductions. The obliviousness condition appears crucial for our proof to work.

AC^0-*Reductions.* We have concentrated on functions computed by TMs with one-way input head(s) as they are provably weaker than polynomial-time functions. However, there are other such classes of functions, e.g., *uniform-*AC^0 or *first-order* functions [8]. Is the AC^0-complete degree conjecture true?

It would also be interesting to know whether the 1-omL-isomorphism conjecture holds.

ACKNOWLEDGMENTS

I thank Professor Somenath Biswas for several suggestions on and corrections at an earlier draft. I am also grateful to Professor Steve Homer

for providing constant encouragement and support. Finally, thanks are due to the anonymous referee whose comments helped enormously in improving the readability of the paper.

REFERENCES

1. M. Agrawal and S. Biswas, Polynomial isomorphism of 1-L-complete sets *in* "Proceedings of the Structure in Complexity Theory Conference, 1993," pp. 75–80.
2. E. Attender, J. Balcázar, and N. Immerman, A first-order isomorphism theorem, *in* "Proceedings of the Symposium on Theoretical Aspects of Computer Science, 1993."
3. E. W. Allender, Isomorphisms and 1-L reductions, *J. Comput. System Sci.* **36**, No. 6 (1988), 336–350.
4. L. Berman, "Polynomial Reducibilities and Complete Sets," Ph.D. thesis, Cornell University, 1977.
5. L. Berman and J. Hartmanis, On isomorphism and density of NP and other complete sets, *SIAM J. Computing* **1** (1977), 305–322.
6. H. Burtschick and A. Hoene, The degree structure of 1-L reductions, *in* "Proceedings of Math. Foundation of Computer Science," Springer Lecture Notes in Computer Science, Vol. 629, pp. 153–161, Springer-Verlag, 1992.
7. S. Fenner, L. Fortnow, and S. Kurtz, The isomorphism conjecture holds relative to an oracle, *in* "Proceedings of FOCS, 1992," pp. 30–39.
8. M. Furst, J. Saxe, and M. Sipser, Parity, circuits and the polynomial time hierarchy, *Math. Systems Theory* **17** (1984), 13–27.
9. K. Ganesin and S. Homer, Complete problems and strong polynomial reducibilities, *in* "Proceedings of the Symposium on Theoretical Aspects of Computer Science," Springer Lecture Notes in Computer Sciences, Vol. 349, pp. 240–250, Springer-Verlag, New York/Berlin, 1988.
10. J. Grollmann and A. Selman, Complexity measures for public-key cryptosystems, *in* "Proceedings of FOCS, 1984," pp. 495–503.
11. J. Hartmanis, On log-tape isomorphisms of complete sets, *Theoret. Comput. Sci.* (1978), 273–286.
12. J. Hartmanis, N. Immerman, and S. Mahaney, One-way log-tape reductions, *in* "Proceedings of FOCS 1978," pp. 65–72.
13. L. A. Hemchandra and A. Hoene, Collapsing degrees via strong computation, *J. Comput. System Sci.* **46** No. 3 (1993), 363–380.
14. N. Immerman, Nondeterministic space is closed under complementation, *SIAM J. Comput.* **17** (1988), 935–938.
15. D. Joseph and P. Young, Some remarks on witness functions for nonpolynomial and noncomplete sets in NP, *Theoret. Comput. Sci.* **39** (1985), 225–237.
16. K. Ko, On some natural complete operators, *Theoret. Comput. Sci.* **37** (1985), 1–30.
17. S. Kurtz, S. Mahaney, and J. Royer, The structure of complete degrees, *in* "Complexity Theory Retrospective" (A. Selman, Ed.), pp. 108–146, Springer-Verlag, New York/Berlin, 1988.
18. S. Kurtz, S. Mahaney, and J. Royer, The isomorphism conjecture fails relative to a random oracle, *in* "Proceedings of STOC, 1989," pp. 157–166.
19. S. Kurtz, S. Mahaney, and J. Royer, Average dependence and random oracles, *in* "Proceedings of the Structure in Complexity Theory Conference, 1992," pp. 306–317.
20. A. L. Selman, A survey of one-way functions in complexity theory, *Math. Systems Theory* **25** (1992), 203–221.
21. L. J. Stockmeyer, "The Complexity of Decision Problems in Automata Theory and logic," Ph.D. thesis, Massachusetts Institute of Technology, 1974.
22. R. Szelepcsényi, The method of forced enumeration for nondeterministic automata, *Acta Informatica* **26** (1988), 279–284.
23. O. Watanabe, On one-one polynomial time equivalence relations, *Theoret. Comput. Sci.* **38** (1985), 157–165.